SIEMENS

PATENT
Attorney Docket No. 2003P06167WOUS

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re: Application of: Group Art Unit: 2121

Applicant: Elmar Thurner Examiner: Gami, Tejal

Serial No.: 10/560,839 Atty. Docket: 2003P06167WOUS

Filed: June 8, 2006 Confirmation No.: 6878

Title: DEVICE AND METHOD FOR PROGRAMMING AND/OR EXECUTING PROGRAMS FOR INDUSTRIAL AUTOMATION SYSTEMS

Mail Stop Appeal Brief - Patent
COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, VA 22313-1450

Sir:

APPELLANT'S BRIEF UNDER 37 CFR 41.37

This brief is in furtherance of the Notice of Appeal filed in this application on April 10, 2008.

1. REAL PARTY IN INTEREST - 37 CFR 41.37(c)(1)(i)

The real party in interest in this Appeal is the assignee Siemens Aktiengesellschaft.

2. RELATED APPEALS AND INTERFERENCES - 37 CFR 41.37(c)(1)(ii)

There is no other appeal, interference or judicial proceeding that is related to or that will directly affect, or that will be directly affected by, or that will have a bearing on the Board's decision in this Appeal.

3. STATUS OF CLAIMS - 37 CFR 41.37(c)(1)(iii)

Claims cancelled: 1-24, 26, 32-33, 39 and 42

Claims withdrawn but not cancelled: None.

Claims pending: 25, 27-31, 34-38, 40-41, and 43-46.

Claims allowed: None.

Claims rejected: 25, 27-31, 34-38, 40-41, and 43-46.

Claims objected to: None.

The claims on appeal are 25, 27-31, 34-38, 40-41, and 43-46.

4. STATUS OF AMENDMENTS - 37 CFR 41.37(c)(1)(iv)

A response was filed on February 8, 2008 under 37 CFR 1.116 subsequent to the final Office Action mailed December 11, 2007. An Advisory Action was issued by the Office on February 28, 2008 maintaining the final rejection of all claims. However, in the Advisory Action, the Examiner entered the Appellant's Amendment and Response of February 8, 2008 for purposes of appeal. Thus, the currently pending claims are those presented in Appellant's February 8, 2008 Amendment and Response.

5. SUMMARY OF THE CLAIMED SUBJECT MATTER- 37 CFR 41.37(c)(1)(v)

Industrial automation systems generally control and/or regulate automated processes or units, such as the automated manufacturing of components. The present invention relates generally to a method, device, and computer program for programming and/or executing programs for such industrial automation systems. *See* the Abstract of the present published application. Advantageously, the present invention enables programs to be easily added, added incrementally, modified, or deleted from all components of an automation system. *See* p. 4, lines 1-22 of paragraph [0041] of the present published application.

Independent Claim 25 is directed to a method for executing a program for an industrial automation system. The method comprises providing a computer unit with input aids, output aids and a display device, having modules and functions respectively representing sub-tasks of an automation solution being modeled and/or created using the input aids and optionally the display device. *See* p. 1, lines 1-7 of paragraph [0003]; p. 2, lines 1-6 of paragraph [0027] and engineering device 22 of Fig. 1 of the present published application. The modules and functions

are structured and networked using the input aids and optionally the display device so as to form at least one hierarchical tree as a machine-independent program. *See* p. 1, lines 9-13 of paragraph [0003] of the present published application. In the method, the machine-independent program is first loaded in the form of the at least one hierarchical tree into the corresponding components of the automation system. *See* p. 1, lines 1-10 of paragraph [0035] of the present published application. The corresponding components of the automation system then execute the machine-independent program present in the form of the at least one hierarchical tree with the aid of at least one object machine assigned to the corresponding components of the automation system. *See* p. 3, paragraphs [0036]-[0037] of the present published application.

The at least one object machine provides operators and objects from which the machine-independent program is provided in the form of the at least one hierarchical tree. *See* p. 3, lines 4-14 of paragraph [0038] of the present published application. During or after the loading the machine-independent program in the form of the at least one hierarchical tree into the corresponding components of the automation system, the operators are instantiated (using the at least one object machine) into corresponding components of the automation system and the symbolic representation of the hierarchical tree is converted to physical addresses to generate a loadable program in the form of an executable program or operator tree. *See* p. 3, lines 1-11 of paragraph [0037] of the present published application.

Independent Claim 38 is directed to a device for executing a program for an industrial automation system. The device comprises at least one computer unit with input aids, output aids and a display device and a component for modelling and/or creating modules and functions, which respectively represent the sub-tasks of an automation solution. *See* p. 1, lines 1-7 of paragraph [0003]; p. 2, lines 1-6 of paragraph [0027] and engineering device 22 of Fig. 1 of the present published application. In addition, the device comprises a component for structuring the modules and functions (and for networking the same) to form at least one hierarchical tree as at least one machine-independent program. *See* p. 1, lines 9-13 of paragraph [0003] of the present published application. The device further comprises a component to load the machine-independent program in the form of the at least one hierarchical tree into the corresponding components of the automation system. *See* p. 1, lines 1-10 of paragraph [0035] of the present published application. The corresponding components of the automation system execute the machine-independent program present in the form of the at least one hierarchical tree. *See* p. 3,

lines 1-8 of paragraph [0036] of the present published application. In the device, the at least one object machine is assigned to the corresponding components of the automation system to execute the machine-independent programs and provides operators and objects from which the machine-independent program is provided in the form of the hierarchical tree. *See* p. 3, lines 4-14 of paragraph [0038] and Figs. 3 and 5 of the present published application. The device still further includes a component to instantiate the operators using the at least one object machine during or after the loading of the machine-independent program into corresponding components of the automation system. *See* p. 3, lines 1-11 of paragraph [0037] of the present published application. Lastly, the device includes a component to convert the symbolic representation of the at least one hierarchical tree to physical addresses to generate a loadable program in the form of an executable program or operator tree. *See* p. 3, lines 1-11 of paragraph [0037] of the present published application.

Independent Claim 47 is directed to a computer program for implementing the method of Claim 25 set forth above. Specification citations provided for claim 25 apply equally to claim 47.

Dependent claims 34 and 44 further specify that the object machine of the method of Claim 25 and the device of Claim 38 respectively are each implemented as a function unit that is closed and processes the at least one hierarchical tree to a runtime system of the automated system. As set forth at p. 2, lines 9-12 of paragraph [0026] of the present published application, "[t]he programs created in the engineering device 22 are transmitted via an information path 25 to the runtime system 23 of the MES device or ERP (Enterprise Resource Planning) device or another destination system." As is also set forth on p. 4, lines 3-7 of paragraph [0041] of the present invention, "[a] program generated with the aid of the invention can be executed on all components of an automation system, on which the object machine or real-time machine is implemented. The program can be adjusted to the runtime."

Dependent claims 37 and 46 further require that the objects of the machine-independent program present as a hierarchical object or operator tree are assigned a collection of infrastructure services or infrastructure functions that access the objects 27 via containers 32 assigned to the objects 27 such that an infrastructure service or an infrastructure function can be used by all the objects 27. See Figs. 3 and 5 of the present published application. As set forth at p. 3, lines 1-23 of paragraph [0032] and Fig. 3 of the present published application, each object

27 has a container 32 hatched on the right side of the object 27 that represents an encapsulating layer about the object 27 and may provide infrastructure functions, such as data networking, data storage, and data visualization in a standard manner for all types of objects. These infrastructure services or corresponding functions are accessed via the container 32 and such access is the same for all objects 27 in the hierarchical tree 33. *See* p. 3, lines 1-23 of paragraph [0032] of the present published application. Thus, advantageously, "[a]n infrastructure service that has been implemented can be used by all objects in the same manner." *See* p. 3, lines 24-26 of paragraph [0032] of the present published application.

6. GROUNDS OF REJECTION TO BE REVIEWED UPON APPEAL - 37 CFR 41.37(c)(1)(vi)

A. Whether Claims 25, 27-31, 34-38, 40-41, and 43-47 stand rejected under 35 U.S.C. 102(e) as being anticipated by U.S. Published Patent Application No. 2006/0259157 to Lo et al. ("Lo").

7. ARGUMENT 37 CFR 41.37(c)(1)(vii)

A. Claims 25, 27-31, 34-38, 40-41, and 43-47 stand rejected under 35 U.S.C. 102(e) as being anticipated by Lo.

(i) Arguments applicable to Independent Claim 25

Independent Claim 25 requires, in part, "converting the symbolic representation of the hierarchical tree to physical addresses to generate a loadable program in the form of an executable program or operator tree." Per MPEP 2131, "[a] claim is anticipated only if each and every element as set forth in the claim is found, either expressly or inherently described, in a single prior art reference." *Verdegaal Bros. v. Union Oil Co. of California*, 814 F.2d 628, 631, 2 USPQ2d 1051, 1053 (Fed. Cir. 1987). Appellant submits that independent Claim 25 is not anticipated by Lo because Lo does not expressly or inherently teach or suggest "converting the symbolic representation of the hierarchical tree to physical addresses to generate a loadable program in the form of an executable program or operator tree" as required in Claim 25.

Lo is directed to providing a plurality of engineering tools at a centralized web server, having the user write programming code using the tools, and downloading the code to a programmable controller. In this way, a plurality of users may write programming code using

the same version of software, for example, and may more easily work on collaborative projects. At lines 1-7 of paragraph [0048] of Lo, Lo discloses engineering tools for generating programming code that reside on a web server and are capable of operating on a web-client in a browser application. A user accesses the web server over a network with a client device, i.e. personal computer, to access the engineering tool. *See* lines 16-22 of paragraph [0045] of Lo. The engineering tool is web-enabled and, in its most basic form, comprises an editor or editors that may be run on the web-browser by the user. *See* lines 14-18 of paragraph [0048] of Lo. According to Lo, at lines 1-6 of paragraph [0050], the transmission between the client and the server is preferably in the form of an XML document. Regarding any steps after generating the code, at lines 1-7 of paragraph [0057], Lo discloses that "[a]fter the code is created, debugged, and compiled, it is downloaded to a programmable controller...and...in one embodiment... programming a program controller comprises two steps: establishing communication between the controller and the server...and downloading the programming code to the controller over the network."

At lines 1-3 of paragraph [0082] of Lo, Lo discloses that its invention "may also comprise a configuration editor 9100 for generating code or [configuration] data for configuring I/O networks and devices." Further, according to Lo, the "[c]onfiguration data often comprise information such as...information relating to the mapping of logical to physical I/O." See lines 18-20 of paragraph [0082] of Lo. The Examiner has maintained that the above disclosures at paragraph [0082] teach "converting the symbolic representation of the hierarchical tree [(XML documents)] to physical addresses to generate a loadable program in the form of an executable program or operator tree." Appellant submits that the Examiner's characterization of Lo is incorrect and that Lo fails to teach or suggest "converting the symbolic representation of the hierarchical tree [(XML documents)] to physical addresses to generate a loadable program in the form of an executable program or operator tree."

Assuming arguendo for the purpose of this argument only that the XML documents are in the form of hierarchical trees, Lo does not teach or suggest converting the symbolic representation of the XML documents to physical addresses to generate a loadable program in the form of an executable program or operator tree through its disclosure of the mapping of logical to physical I/O. As set forth above, at paragraph [0050] of Lo, Lo discloses the transmission between the client and the server is preferably in the form of an XML document.

Importantly, Lo does not express or inherently describe converting the symbolic representation of this XML document, let alone converting the symbolic representation of this XML document to physical addresses to generate a loadable program in the form of an executable program or operator tree. Thus, because Lo fails to expressly or inherently describe each and every element as set forth in independent Claim 25, the rejection of Claim 25 under 35 U.S.C. 102(e) is not supported by the prior art and must be withdrawn.

(ii) Arguments applicable to Independent Claim 38

Independent Claim 38 includes a similar limitation to Claim 25 in structural form, namely "a component to convert the symbolic representation of the at least one hierarchical tree to physical addresses to generate a loadable program in the form of an executable program or operator tree." For the reasons set forth above with respect to Claim 25, Claim 38 is not anticipated by Lo because Lo also does not expressly or inherently describe converting the symbolic representation of this XML document, let alone converting the symbolic representation of this XML document to physical addresses to generate a loadable program in the form of an executable program or operator tree. Accordingly, Lo fails to expressly or inherently describe each and every element as set forth in independent Claim 38. Thus, the rejection of Claim 38 is also not supported by the prior art and must be withdrawn.

(iii) Arguments applicable to Independent Claim 47

Independent Claim 47 requires a computer program implementing the method of Claim 25. Similar to independent Claim 25, independent Claim 47 requires that the implemented method comprises "converting the symbolic representation of the hierarchical tree to physical addresses to generate a loadable program in the form of an executable program or operator tree." As set forth above with respect to Claim 25, Lo does not does not express or inherently describe converting the symbolic representation of this XML document, let alone converting the symbolic representation of this XML document to physical addresses to generate a loadable program in the form of an executable program or operator tree. Thus, because Lo fails to expressly or inherently describe each and every element as set forth in independent Claim 47, the rejection of Claim 47 is not supported by the prior art and must be withdrawn.

(iv) Arguments applicable to Dependent Claims 34 and 44

Dependent claims 34 and 44 require that "the object machine is implemented as a function unit that is closed and that processes...at least one hierarchical tree to a runtime system

of the automated system." Dependent claims 34 and 44 provide further reason for allowance because Lo is wholly silent as to an "object machine that is implemented as a function unit that is closed and that processes...at least one hierarchical tree to a runtime system of the automated system." The Examiner has maintained that paragraphs [0047] and [0048] of Lo disclose "running in an application making them operating system independent." *See e.g.* page 11 of the December 11, 2007 Office Action.

First, Appellant submits that the Examiner has never clearly articulated a reason for rejecting dependent claims 34 and 44 in the prosecution of the present application. MPEP §706 specifically states that "[t]he goal of examination is to clearly articulate any rejection early in the prosecution process so that the applicant has the opportunity to provide evidence of patentability and otherwise reply completely at the earliest opportunity." (emphasis added). 37 CFR 1.104 (c)(2) further requires that "[t]he pertinence of each reference, if not apparent, must be clearly explained and each rejected claim specified."

Specifically, in the December 11, 2007 Office Action, the Examiner's reasons for the rejection of Claims 34 and 44 were as follows:

> -As to dependent Claim 34, Lo teaches the method according to Claim 25, wherein the object machine is implemented as a function unit that is closed and that processes the hierarchical tree to a runtime (see paragraphs [0047]-[0048] where XML documents have the structure of hierarchical trees). *See* page 8 of the December 11, 2007 Office Action.
>
> -As to dependent Claim 44, Lo teaches the device according to Claim 25, wherein the object machine is implemented as a function unit that is closed and that processes the hierarchical tree to a runtime (see paragraphs [0047]-[0048] where XML documents have the structure of hierarchical trees). *See* page 10 of the December 11, 2007 Office Action.
>
> -Examiner is not persuaded. The claims and only the claims form the metes and bounds of the invention. See paragraphs [0047] and [0048] where Lo teaches running in an application making them operating system independent. *See* the Response to Arguments portion of the December 11, 2007 Office Action.

Appellant still remains unsure as to how "running in an application making them operating system independent" can be construed as an "object machine [that] is implemented as a function unit that is closed and that processes...at least one hierarchical tree to a runtime system of the automated system." Because a clear line of reasoning for the rejection of claims 34 and 44 has never been provided sufficient for Appellant to respond to the argument, Appellant submits

the Examiner has not presented a *prima facie* case of anticipation for dependent claims 34 and 44 and that the rejections of dependent claims 34 and 44 must be withdrawn.

Second, Appellant submits that Lo is wholly silent as to an "object machine [that] is implemented as a function unit that is closed and that processes the hierarchical tree to a runtime system of the automated system." As set forth in paragraph [0048] of Lo, Lo discloses engineering tools for generating programming code that reside on a web server and are capable of operating on a web-client in a browser application. *See* lines 3-7 of paragraph [0048] of Lo. A user accesses the web server over a network with a client device, i.e. personal computer, to access the engineering tools. *See* lines 16-22 of paragraph [0045] of Lo. According to Lo, "the engineering tools are configured to be operated with an HTML or XML interface so that they can operate on any available browser." *See* lines 14-16 of paragraph [0047] of Lo. Even *assuming arguendo* for the purpose of this argument only that the programming code is in the form of at least one hierarchical tree as claimed and that the programmable logic controller (PLC) of Lo is an automated system, Lo fails to describe an "object machine [that] is implemented as a function unit that is closed and that processes the hierarchical tree (the XML documents) to a runtime system of the automated system (PLC)." Accordingly, the prior art does not support the rejection of dependent claims 34 and 44 under 35 U.S.C. 102(e) and must be withdrawn.

(v) Arguments applicable to Dependent Claims 37 and 46

Dependent claims 37 and 46 require that "the objects of the machine-independent program present as a hierarchical object or operator tree are assigned a collection of infrastructure services or infrastructure functions that access the objects via containers assigned to the objects such that an infrastructure service or infrastructure function can be used by all the objects." In this way, "[i]nfrastructure services or corresponding functions are accessed via the container 32 and such access is the same for all objects in the hierarchical tree." *See* lines 18-21 of paragraph [0033] of the present published application. The Examiner points to paragraph [0064] of Lo and Lo's disclosure of ACTIVEX controls and contends that it was well-known at the time the invention was made that ACTIVEX is Microsoft technology used for developing reusable object oriented software components. Lines 11-14 of paragraph [0064] of Lo specifically discloses that "Microsoft...provides software for allowing applications running on its Internet Explorer® to access client side system resources by means of ACTIVEX controls embedded in the application." This, according to paragraph [0064] of Lo, allows the application

to be installed on the client's device. *See* lines 4-6 of paragraph [0064] of Lo. Even *assuming arguendo* for the purpose of this argument only that Lo includes containers and that the applications referred to by Lo in paragraph [0064] include a hierarchical tree (i.e. an XML document), nothing in Lo discloses assigning the hierarchical tree (i.e. an XML document) a collection of infrastructure services or infrastructure functions that access objects via the containers (i.e. ACTIVEX controls). Moreover, as with dependent claims 34 and 44, the Examiner has not articulated any clear reasoning as to how Lo could be construed to read on the present claims. Accordingly, Appellant submits that Lo also fails to support the rejection of dependent claims 37 and 46 under 35 U.S.C. 102(e) and the rejection of claims 37 and 46 must be withdrawn.

8. CLAIMS APPENDIX - 37 CFR 41.37(c) (1) (viii).

A copy of the claims involved in this appeal is attached as a claims appendix under 37 CFR 41.37(c) (1) (viii).

9. EVIDENCE APPENDIX - 37 CFR 41.37(c) (1) (ix)

None is required under 37 CFR 41.37(c) (1) (ix).

10. RELATED PROCEEDINGS APPENDIX - 37 CFR 41.37(c) (1) (x)

None is required under 37 CFR 41.37(c) (1) (x).

Respectfully submitted,

Dated: 6/4/08

By: _____

John P. Musone
Registration No. 44,961
(407) 736-6449

Siemens Corporation
Intellectual Property Department
170 Wood Avenue South
Iselin, New Jersey 08830

APPENDIX OF CLAIMS ON APPEAL

25. A method for executing a program for an industrial automation system, comprising:

providing a computer unit with input aids, output aids and a display device, having modules and functions respectively representing sub-tasks of an automation solution being modeled and/or created using the input aids and optionally the display device, having the modules and functions being structured and networked using the input aids and optionally the display device as to form at least one hierarchical tree as a machine-independent program,

loading the machine-independent program in the form of the at least one hierarchical tree into the corresponding components of the automation system, wherein the corresponding components of the automation system execute the machine-independent program present in the form of the at least one hierarchical tree with the aid of at least one object machine assigned to the corresponding components of the automation system, and wherein the at least one object machine provides operators and objects from which the machine-independent program is provided in the form of the at least one hierarchical tree; and

during or after loading of the machine-independent program, instantiating the operators using the at least one object machine into corresponding components of the automation system; and

converting the symbolic representation of the hierarchical tree to physical addresses to generate a loadable program in the form of an executable program or operator tree.

27. The method according to claim 25, wherein the objects of the machine-independent program are present in the form of at least one hierarchical object or operator tree in the corresponding components of the automation system and are processed interpretatively.

28. The method according to claim 27, wherein the machine-independent program is present in the form of at least one object or operator tree with a structure equivalent or similar to the representation of the program in the display device.

29. The method according to claim 25, wherein the machine-independent program is loaded into the corresponding components of the automation system using a machine-independent, symbolic representation of the hierarchical tree.

30. The method according to claim 29, wherein the machine-independent and symbolic representation of the hierarchical tree is in the form of a byte code or a markup language such as extended markup language.

31. The method according to claim 25, wherein the object machine is configured as a real-time object machine with deterministic response and cycle times.

34. The method according to claim 25, wherein the object machine is implemented as a function unit that is closed and that processes the at least one hierarchical tree to a runtime system of the automated system.

35. The method according to claim 27, wherein the object machine is implemented in a distributed manner as at least one object, with the hierarchical object or operator tree processing itself.

36. The method according to claim 25, wherein the modules and functions are assigned model information and/or meta-information using the input aids and/or the display device.

37. The method according to claim 27, wherein the objects of the machine-independent program present as a hierarchical object or operator tree are assigned a collection of infrastructure services or infrastructure functions that access the objects via containers assigned to the objects such that an infrastructure service or an infrastructure function can be used by all the objects.

38. A device for executing a program for an industrial automation system, comprising:

at least one computer unit with input aids, output aids and a display device;

a component for modeling and/or creating modules and functions, which respectively represent the sub-tasks of an automation solution;

a component for structuring the modules and functions and for networking the same, to form at least one hierarchical tree as at least one machine-independent program; and

a component to load the machine-independent program in the form of the at least one hierarchical tree into the corresponding components of the automation system with the corresponding components of the automation system executing the machine-independent program present in the form of the at least one hierarchical tree, wherein at least one object machine is assigned to the corresponding components of the automation system to execute the machine-independent programs, and wherein the at least one object machine provides operators and objects from which the machine-independent program is provided in the form of the hierarchical tree;

a component to instantiate the operators using the at least one object machine during or after the loading of the machine-independent program into corresponding components of the automation system; and

a component to convert the symbolic representation of the at least one hierarchical tree to physical addresses to generate a loadable program in the form of an executable program or operator tree.

40. The device according to claim 38, wherein the machine-independent program is present in the form of at least one object or operator tree with a structure equivalent or similar to the representation of the program in the display device.

41. The device according to claim 38, wherein the at least one object machine is configured as a real-time object machine with deterministic response and cycle times.

43. The device according to claim 38, further comprising a device for assigning model information and/or meta-information to the modules and functions.

44. The device according to claim 38, wherein the object machine is implemented as a function unit that is closed and processes the at least one hierarchical tree to a runtime system of the automated invention.

45. The device according to claim 38, wherein the object machine is implemented in a distributed manner as at least one object, with the hierarchical object or operator tree processing itself.

46. The device according to claim 38, wherein the objects of the machine-independent program present as a hierarchical object or operator tree are assigned a collection of infrastructure services or infrastructure functions that access the objects via containers assigned to the objects such that an infrastructure service or infrastructure function can be used by all the objects.

47. A computer program implementing a method for executing a program for an industrial automation system, comprising:

providing a computer unit with input aids, output aids and a display device, having modules and functions respectively representing sub-tasks of an automation solution being modeled and/or created using the input aids and optionally the display device, having the modules and functions being structured and networked using the input aids and optionally the display device so as to form a hierarchical tree as a machine-independent program;

loading the machine-independent program in the form of the hierarchical tree into the corresponding components of the automation system, wherein the corresponding components of the automation system execute the machine-independent program present in the form of the hierarchical tree with the aid of at least one object machine assigned to the corresponding components of the automation system, and wherein the at least one object machine provides operators and objects from which the machine-independent program is provided in the form of the hierarchical tree;

during or after loading of the machine-independent program, instantiating the operators using the at least one object machine into corresponding components of the automation system; and

converting the symbolic representation of the hierarchical tree to physical addresses to generate a loadable program in the form of an executable program or operator tree.

EVIDENCE APPENDIX

None.

RELATED PROCEEDINGS APPENDIX

None.